Engenharia de Software I

UniSenac Pelotas

Profª Bruna Ribeiro

email: brgribeiro@senacrs.com.br

Plano de ensino

Caracterização da unidade curricular

- Modelagem e especificação de necessidades a serem atendidas por um sistema de software na abordagem de análise orientada a objetos.

Elementos de competência

- Compreender os papéis envolvidos no desenvolvimento de software;
- Compreender as principais técnicas de levantamento de requisitos;
- Propor abstrações coerentes para um software de acordo com o domínio de problema abordado;
- Construir diagramas utilizando corretamente a especificação da UML.

Bases tecnológicas

- Introdução à Engenharia de Software.
- Compreensão dos papéis envolvidos no desenvolvimento de software.
- O papel do Analista de Sistemas.
- Modelos de Processo de Software.
- Identificação do problema: técnicas e artefatos relacionados.
- Requisitos funcionais e não funcionais.
- Definição de personas.
- Histórias do usuário.
- Casos de uso.
- UML e artefatos em nível de análise.

Bibliografia básica

- FERNANDES, João M. e MACHADO, Ricardo J. Requisitos em projetos de software e de sistemas de informação. Novatec. Edição 1. 2017.
- GUEDES, Gilleanes T. A. UML 2 - Uma Abordagem Prática. Novatec. Edição 3. 2018.
- DEBASTIANI, Carlos Alberto. Definindo Escopo em Projetos de Software. Novatec. Edição 1. 2015.

Bibliografia complementar

- BARNES, David J.; KÖLLING, Michael. Programação orientada a objetos com Java. São Paulo: Pearson Prentice Hall, 2004.
- BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. UML: guia do usuário. 2. ed. rev. atual. Rio de Janeiro: Elsevier, 2012.
- LARMAN, Craig. Utilizando UML e padrões: uma introdução à análise e ao projeto orientados a objetos e ao processo unificado. 3. ed. Porto Alegre: Bookman, 2007.
- PAULA FILHO, Wilson de Pádua. Engenharia de software: fundamentos, métodos e padrões. 3. ed. Rio de Janeiro: LTC, 2009.
- RUMBAUGH, James et al. Modelagem e projetos baseados em objetos. Rio de Janeiro: Elsevier, 2004.

Avaliação

- Mínimo 75% de presença
- Entregas pontuais
- Atividade Avaliativa Individual
- Projeto Final Integrado
  - Engenharia de Software
  - Desenvolvimento Web para Dispositivos Móveis
  - Programação Web

Dúvidas

- Dúvidas????

Engenharia de Software I

Introdução

UniSenac Pelotas

Introdução

- Engenharia de Software é uma área da computação voltada à **especificação**, **desenvolvimento** e **manutenção** de sistemas de software, com aplicação de tecnologias e práticas de gerência de projetos, visando **organização**, **produtividade** e **qualidade**.

Principais Pontos

- Desenvolvimento Sistematizado:
  - Envolve o uso de processos bem definidos e estruturados, que incluem etapas como análise de requisitos, design, implementação, testes e manutenção. Esses processos ajudam a garantir que o software seja desenvolvido de maneira organizada e que atenda às necessidades dos usuários finais.
- Gestão de Projetos:
  - Envolve o planejamento, organização e controle dos recursos e atividades necessárias para completar o desenvolvimento de software dentro do prazo. Isso inclui a definição de cronogramas, alocação de tarefas, gerenciamento de riscos e monitoramento do progresso.

Principais Pontos

- Qualidade e Manutenção
  - A Engenharia de Software coloca grande ênfase na criação de software de alta qualidade, que seja fácil de manter e evoluir.
- Colaboração e Comunicação:
  - O desenvolvimento de software normalmente envolve equipes multidisciplinares, e a Engenharia de Software fornece práticas e ferramentas para facilitar a comunicação e colaboração entre os membros da equipe, incluindo desenvolvedores, designers, gerentes de projeto, testadores, e clientes.

Objetivos da Engenharia de Software

- Desenvolvimento de software de qualidade
- Satisfação do cliente
- Entrega no prazo e no orçamento
- Redução de riscos
- Eficiência e produtividade
- Documentação adequada
- Padronização

Engenharia de Software

- A Engenharia de Software desempenha um papel crucial no desenvolvimento de um projeto de software, pois ela fornece as diretrizes, práticas e metodologias necessárias para garantir que o software seja criado de forma eficiente e com alta qualidade.

Importância da engenharia de software

- Garantia de Qualidade
  - A Engenharia de Software utiliza práticas como revisões de código, testes automatizados e validação de requisitos para minimizar erros e defeitos. Isso é fundamental para garantir que o software funcione corretamente e atenda às necessidades dos usuários.
- Manutenção
  - O uso de padrões de design e práticas de codificação limpa facilita a manutenção e evolução do software ao longo do tempo, permitindo que novas funcionalidades sejam adicionadas sem comprometer a qualidade existente.

Importância da engenharia de software

- Divisão do Projeto em Módulos
  - A Engenharia de Software promove a decomposição do sistema em módulos ou componentes menores e mais gerenciáveis, facilitando o desenvolvimento, teste e manutenção.
- Alocação Eficiente de Recursos
  - O gerenciamento eficaz de recursos humanos e técnicos é facilitado por práticas de Engenharia de Software, garantindo que cada membro da equipe seja utilizado da melhor maneira possível.

Importância da engenharia de software

- Planejamento e Gerenciamento de Projetos
  - A Engenharia de Software inclui ferramentas e metodologias para o planejamento detalhado do projeto, gerenciamento de riscos, estimativas de tempo e custo, e monitoramento contínuo do progresso. Isso ajuda a garantir que o projeto seja concluído dentro do prazo e orçamento estabelecidos.
- Atendimento aos Requisitos
  - A Engenharia de Software assegura que os requisitos do cliente sejam claramente definidos, compreendidos e implementados. Isso reduz o risco de desentendimentos e garante que o produto final atenda às expectativas do cliente.

Importância da engenharia de software

- Flexibilidade para Mudanças
  - Muitas vezes, os requisitos de software evoluem durante o desenvolvimento. A Engenharia de Software incorpora práticas que permitem a adaptação às mudanças de forma controlada, sem comprometer a qualidade do projeto.
- Documentação e Controle
  - A manutenção de documentação detalhada e o uso de sistemas de controle de versão ajudam a gerenciar as mudanças e preservar a integridade do software ao longo do ciclo de vida do projeto.

Importância da engenharia de software

- A Engenharia de Software é fundamental para o sucesso de um projeto de software. Ela garante que o desenvolvimento seja conduzido de maneira organizada, eficiente e dentro dos parâmetros de qualidade, prazo e custo.
- Além disso, ela promove a satisfação do cliente e a sustentabilidade do software a longo prazo, assegurando que o projeto não apenas atenda às necessidades atuais, mas também seja capaz de evoluir para atender às demandas futuras.

Evolução

- Primeira Era (1950 - 1960)
  - O conceito de software começou a ganhar importância com o desenvolvimento dos primeiros computadores eletrônicos. O foco estava em escrever código diretamente em linguagem de máquina ou linguagem assembly.
  - Desafio:
    - A programação era feita sem metodologias estruturadas. Erros eram comuns e difíceis de corrigir. O software era visto como um acessório ao hardware.

Evolução

- Segunda Era: A crise do Software (1960 - 1970)
  - À medida que os sistemas de software se tornaram mais complexos e críticos, surgiram problemas significativos: atrasos nos prazos, estouros de orçamento, software de baixa qualidade e falhas catastróficas.
  - A "Crise do Software" levou à necessidade de metodologias mais estruturadas para o desenvolvimento de software, dando origem ao campo da Engenharia de Software

Evolução

- Terceira Era: Metodologias Estruturadas (1970 - 1980)
  - Surgimento de metodologias como o Modelo Cascata (Waterfall), que organizava o desenvolvimento de software em etapas sequenciais: requisitos, design, implementação, testes e manutenção.
  - Engenharia de Requisitos:
    - Início da formalização do processo de levantamento de requisitos e design de sistemas, reconhecendo a importância de capturar corretamente as necessidades dos usuários antes de iniciar o desenvolvimento.

Evolução

- Quarta Era: Reuso (1980 - 1990)
  - A reutilização de componentes de software começou a ser promovida como uma maneira de aumentar a produtividade e melhorar a qualidade, levando ao desenvolvimento de bibliotecas e frameworks.

Evolução

- Quinta Era: Metodologias Ágeis e Desenvolvimento Iterativo (1990 - 2000)
  - Em resposta à rigidez dos modelos tradicionais como o Cascata, surgiram metodologias ágeis (ex.: Scrum, XP) que enfatizam entregas incrementais e iterativas, colaboração contínua com o cliente, e flexibilidade para mudanças nos requisitos.

Evolução

- Sexta Era (2000 - Presente)
  - Práticas de Integração Contínua e Entrega Contínua se tornaram comuns, permitindo que equipes integrem e testem mudanças de código continuamente, resultando em ciclos de desenvolvimento mais rápidos e software de melhor qualidade.

Falhas

- Geralmente os projetos de desenvolvimento de software falham devido às seguintes causas:
  - Requisitos mal definidos ou mal compreendidos;
  - Mudança nos requisitos;
  - Problemas na comunicação;
  - Complexidade subestimada;
  - Testes insuficientes;
  - Entre outros.

Metodologias de desenvolvimento

- Através de uma **metodologia** de desenvolvimento de software podemos tratar essas causas
- Os sintomas serão eliminados e será mais fácil desenvolver e manter um software de qualidade de forma previsível e que possa ser repetida.

O que é uma metodologia????

Metodologia

- Segundo o dicionário Aurélio:
  - Metodologia é o estudo dos métodos;
  - Caminho pelo qual se atinge um objetivo;
  - Modo de proceder, maneira de agir.

Por que utilizar uma metodologia?

- Organização e Estrutura:
  - Metodologias oferecem um **plano estruturado** para o desenvolvimento de software, dividindo o **processo** em etapas bem **definidas**.
- Melhoria da Eficiência:
  - Metodologias estabelecem processos e práticas que podem **melhorar** a eficiência do desenvolvimento. Elas ajudam a **evitar retrabalho**, alocar recursos de forma eficaz e minimizar a ocorrência de erros.

Por que utilizar uma metodologia?

- Qualidade do Software:
  - Uma metodologia bem escolhida inclui práticas de **teste** e garantia de **qualidade**. Isso resulta em software mais **confiável** e de alta qualidade, com menos defeitos e problemas após a implantação.
- Previsibilidade:
  - Metodologias oferecem uma visão clara do **processo** e dos **prazos**. Isso permite que a equipe faça estimativas realistas e forneça cronogramas confiáveis aos clientes.

Por que utilizar uma metodologia?

- Comunicação e Colaboração:
  - Metodologias incentivam a **comunicação clara** e regular entre os membros da equipe e com os clientes. Isso promove a **colaboração**, ajuda a **resolver problemas** rapidamente e mantém todos envolvidos atualizados sobre o progresso do projeto.
- Redução de Conflitos e Ambiguidade:
  - Uma metodologia claramente definida pode ajudar a evitar **mal-entendidos** e **conflitos** dentro da equipe, já que todos seguem as mesmas diretrizes e práticas estabelecidas.

O que muda?

- O processo de implantação de uma metodologia tende a aumentar o trabalho e a burocracia, tornando o trabalho mais lento;
- A manutenção da documentação pode ser tediosa;
- Porém, a longo prazo os benefícios aparecem e não são poucos. É importante que se escolha a **metodologia certa** para cada **situação**.

Dúvidas

- Dúvidas????

Atividade

- Várias foram as “falhas históricas de desenvolvimento de software” que geraram prejuízos. Faça uma pesquisa e cite exemplos de falhas descritos detalhadamente (mínimo 3 exemplos).
- Qual foi o motivador para o surgimento da Engenharia de Software?
- Em duplas, monte uma linha do tempo sobre a evolução da engenharia de software desde os anos 1950 até os dias de hoje.